Sequential Line Search for Efficient Visual Design Optimization by Crowds

YUKI KOYAMA, ISSEI SATO, DAISUKE SAKAMOTO, and TAKEO IGARASHI, The University of Tokyo

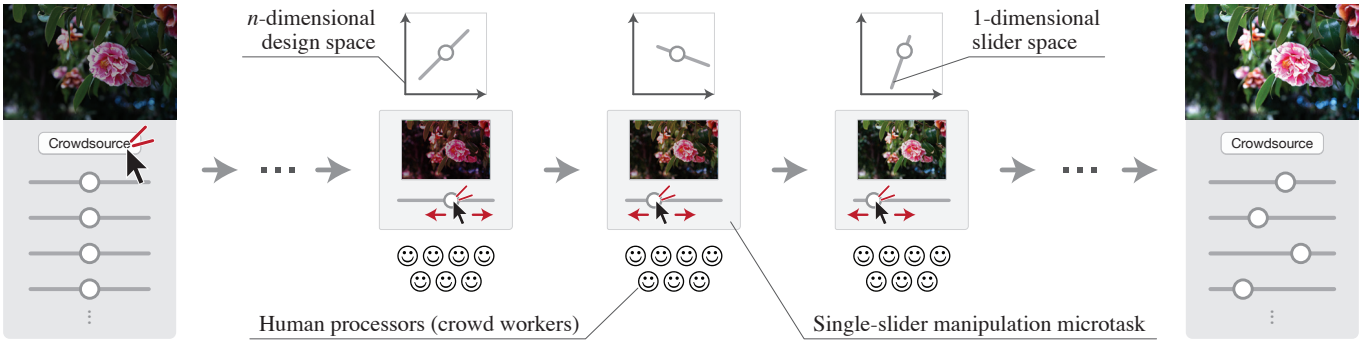

Fig. 1. **Concept of our method.** We envision that design software equips a "Crowdsource" button for running the crowd-powered search for the slider values that provides perceptually "best" design. To enable this, we present a novel extension of *Bayesian optimization*, where the system decomposes the n-dimensional optimization problem into a sequence of one-dimensional line search queries that can be solved by crowdsourced human processors.

Parameter tweaking is a common task in various design scenarios. For example, in color enhancement of photographs, designers tweak multiple parameters such as "brightness" and "contrast" to obtain the best visual impression. Adjusting one parameter is easy; however, if there are multiple correlated parameters, the task becomes much more complex, requiring many trials and a large cognitive load. To address this problem, we present a novel extension of *Bayesian optimization* techniques, where the system decomposes the entire parameter tweaking task into a sequence of one-dimensional *line search* queries that are easy for human to perform by manipulating a single slider. In addition, we present a novel concept called *crowd-powered visual design optimizer*, which queries crowd workers, and provide a working implementation of this concept. Our *single-slider manipulation* microtask design for crowdsourcing accelerates the convergence of the optimization relative to existing comparison-based microtask designs. We applied our framework to two different design domains: photo color enhancement and material BRDF design, and thereby showed its applicability to various design domains.

CCS Concepts: • **Computing methodologies** → **Computer graphics**;

Additional Key Words and Phrases: Bayesian optimization, crowdsourcing, human computation, computational design

ACM Reference format:
Yuki Koyama, Issei Sato, Daisuke Sakamoto, and Takeo Igarashi. 2017. Sequential Line Search for Efficient Visual Design Optimization by Crowds. *ACM Trans. Graph.* 36, 4, Article 48 (July 2017), 11 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073598

1 INTRODUCTION

Parameter tweaking is a common task in various design scenarios. For example, in color enhancement of photographs, designers carefully tweak multiple parameters such as "brightness" and "contrast" to obtain the most visually pleasing results. Similarly, game authoring, presentation slide design, procedural modeling, 3D rendering, *etc.* involve parameter tweaking. However, manual parameter tweaking is tedious and difficult because the user needs to explore a large high-dimensional search space by directly controlling many low-level parameters. The relation between individual parameters and the result is often unknown, so the user needs to proceed by trial and error (*i.e.,* move a slider, see the result, and move the slider back). In this process, the user needs to remember the effect of each parameter during the search, which imposes a significant mental workload. This exploration is combinatorial, making the task even more difficult.

A possible approach is to show multiple candidates and let the user select the best one (*e.g.,* [Marks et al. 1997]); however, it is difficult to cover a high-dimensional search space with a limited number of instances, and the task would still not be easy if the number of candidates is large. Brochu *et al.* [2007] proposed an alternative, where a search task is converted into a *sequence* of simple pairwise comparison tasks. The user only needs to compare two images presented by the system and answer which is better. The user needs to repeat this task multiple times, but this is much easier than dealing with many sliders at once. The system internally formulates the problem as a *Bayesian optimization* and selects candidate images that can most efficiently lead to the best parameter set. In a sense, this is a hybrid human-computer cooperative problem solving approach, where the human performs simple perceptual tasks while the computer guides the human and integrates the data.

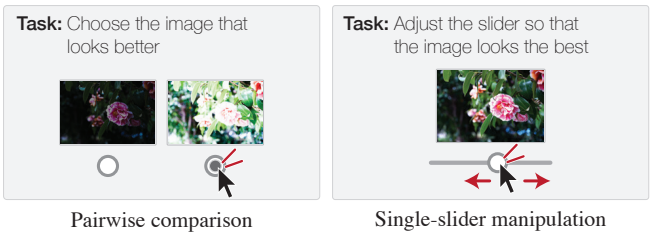

Fig. 2. **Microtask design.** (Left) Existing methods use pairwise comparison microtasks. (Right) We propose to use single-slider manipulation microtasks.

A problem with the above method is efficiency. It requires many iterations (*i.e.,* interactive comparison tasks) to reach an optimum. This is because very limited information (*i.e.,* the relative order of two discrete samples) is obtained from a single iteration. Another limitation is that their implementation is currently limited to a single user. The method converts a complicated compound task into a sequence of simple tasks, so it has a potential to be used in a crowdsourcing setting where many workers complete simple microtasks in parallel. However, this possibility has not been investigated so far.

1.1 Contributions

To address these problems, we propose two extensions of Brochu *et al.*'s method [2007]. First, we propose a Bayesian optimization framework based on *line search* oracles instead of pairwise comparison oracles; our framework decomposes the entire problem into a sequence of one-dimensional slider manipulation tasks (Figure 2). This makes it possible to obtain much richer information in a single iteration compared with a pairwise comparison of discrete samples and to reach the optimum much more efficiently. The difficulty of this task is slightly greater, but it is still comparable to the comparison-based task.

The second extension is to implement an optimization framework using *crowdsourced human computation* instead of having the user in the loop, which enables "semi-automatic" execution of the optimization. (Although crowds interact with the system, the user experience is automatic.) We present the concept of a *crowd-powered visual design optimizer*, and implement within the Bayesian optimization framework based on line search oracles. Once the user submits a high-dimensional design task to the system, it generates a sequence of *single-slider manipulation* microtasks and deploys them to a crowdsourcing platform. Crowd workers complete the tasks independently, and the system gradually reaches the optimal solution. Figure 1 illustrates this concept.

We demonstrate the effectiveness of our crowd-powered optimizer using two different design domains: photo color enhancement with a 6-dimensional design space, and material appearance design based on parametric bidirectional reflectance distribution function (BRDF) models with 3- and 7-dimensional design spaces. We also show that our slider-based method makes the optimization converge faster and yields better solutions than comparison-based methods do, both in a synthetic simulation and in an actual crowdsourcing setting.

1.2 Assumptions

Design domains. We assume that the target parameters are continuous, and thus discrete ones (*e.g.,* font type) are out of the scope. We also assume that the corresponding visual changes continuously with respect to each parameter. From these assumptions, we consider the goodness function to be a continuous, smooth function. We expect that the design space is reasonably parameterized by at most a dozen parameters as in most softwares; the parametrization itself is out of our scope. It is necessary that even novices can assess the goodness of the designs (but they do not need to know how it can be improved). These assumptions are sufficiently general and we believe that our method is applicable to problems in a wide variety of purposes such as photo color enhancement, parametric BRDF design, facial expression modeling using blendshape, 2D graphic design (*e.g.,* posters), procedural texturing (*e.g.,* Perlin noise), and post-rendering image effects.

Crowds. We assume that there exists a common preference shared by the crowd workers. Each crowd worker responds with some "noise" added to this common preference, and thus, gathering responses from a sufficient number of them and averaging the responses should provide a good approximation of the underlying common preference. We do not handle differences in preference among crowd workers; for example, we do not consider demographic factors in preference [Reinecke and Gajos 2014].

2 RELATED WORK

2.1 Bayesian Optimization

Bayesian optimization finds a maximum (or a minimum) of a black-box function. It is especially effective for optimizing expensive-to-evaluate functions, because it tries to minimize the number of iterations, *i.e.,* function evaluations. This technique actively chooses a sampling point in each iteration on the basis of previous observations. Researchers have used this technique, for example, for optimizing hyperparameters in machine learning models such as convolutional neural networks [Snoek et al. 2012]. We recommend that readers refer to the comprehensive introductions [Brochu et al. 2010b; Shahriari et al. 2016] for details.

Because human evaluation is considerably expensive compared to machine evaluation, Bayesian optimization can be a reasonable choice for human-in-the-loop settings. Brochu *et al.* [2007] applied Bayesian optimization techniques to material BRDF design, which requires human evaluation. They used an isotropic monotone BRDF model, yielding a 3-dimensional design space. We also demonstrate the applicability of our method in material BRDF design scenarios with 3-dimensional (monotone) and 7-dimensional (full-color) design spaces. Later, Brochu *et al.* [2010a] applied Bayesian optimization techniques to 4- and 12-dimensional design spaces of fluid animations. To handle such high-dimensional spaces, they incorporated domain-specific data as a prior knowledge.

2.2 Crowdsourced Human Computation

Von Ahn [2005] described the concept of *human computation* in his Ph.D. thesis as *"a paradigm for utilizing human processing power to solve problems that computers cannot yet solve"*. For comprehensive

discussions on the definition of human computation, see the survey by Quinn and Bederson [2011]. Human processors are often employed on demand through *microtask-based crowdsourcing* services, such as Amazon Mechanical Turk[1] and CrowdFlower[2]. Little *et al.* [2010] described the concept of *human computation algorithms*, in which crowdsourced human processors are incorporated into algorithms in the form of function calls. In the computer graphics community, Gingold *et al.* [2012] devised human computation algorithms to solve perceptual computer vision problems. Koyama *et al.* [2014] presented a human computation algorithm for constructing preference models for target design spaces. In this paper, we present a human computation algorithm for design optimization.

Although we believe that we are the first to "explicitly" present the concept of a crowd-powered visual design optimizer, several related attempts have been conducted. Bernstein *et al.* [2011] presented a human computation algorithm called *rapid refinement*, which is used to identify the best frame in a 10-second video. The search space is limited to the video's seekbar. Laursen *et al.* [2016] presented an algorithm to select the best icon set for a GUI design from many icon candidates. They used crowdsourced human computation for gathering perceptual data about candidate icons, but the optimization process itself is a simple exhaustive search and does not involve human computation. Another notable related area is *interactive evolutionary computation* (IEC) for visual design [Clune and Lipson 2011; Sims 1991], in which human evaluation is incorporated into the loop of evolutionary computation, and some studies used crowdsourcing to involve such human input (*e.g.,* [Yu and Nickerson 2011]). Besides the underlying mathematics, these methods and our own differ in several respects. First, most IEC methods are used to explore creative designs, whereas our method is used to determine in an efficient manner a single optimal design from a design space. Second, as our target problem is a continuous optimization, we can utilize continuous line search tasks rather than typical selection or rating tasks used in IEC.

2.3 Microtask Design for Crowdsourcing

To stably obtain data from crowdsourcing, it is important to design the tasks to be micro, *i.e.,* able to be completed in a few minutes, and easy, *i.e.,* able to be conducted by unskilled crowds. One of the most popular microtask designs is pairwise comparison: showing two options and asking the crowd worker to select one according to some criteria [Chaudhuri et al. 2013; Garces et al. 2014; O'Donovan et al. 2014; Secord et al. 2011; Zhu et al. 2014]. This is also called 2-forced alternative comparison (2FAC). Showing more than two options is a straightforward extension of 2FAC, and probabilistic models for this type of comparison data are available [Tsukida and Gupta 2011]. In section 7, we compare these comparison-based methods with our slider-based method and show that ours is more effective for solving optimization problems.

Another popular microtask design is to gather n-point Likert-type scale (*e.g.,* $n = 5$) data for a single stimulus (*e.g.,* [O'Donovan et al. 2011; Serrano et al. 2016; Streuber et al. 2016]). However, we consider that this is not suitable in our problem setting of solving optimization problems; around the optimum, the crowds' responses would always be the highest option, which is not informative for finding the optimum. More importantly, unless crowd workers are familiar with the entire design space, ratings are unlikely to be consistent. Gathering n-point Likert-type scale data for pairwise stimuli comparisons (*e.g.,* [Koyama et al. 2014; Yumer et al. 2015]) is also a possible microtask design; however, probabilistically modeling this data for Bayesian optimization is not trivial. Wilber *et al.* [2014] proposed a microtask design of selecting k options from n options ($k < n$) for the purpose of similarity-driven embedding. In summary, while most of the existing microtask designs involve only discrete samples, our slider-based microtask design involves continuous searches, which provide richer information.

2.4 Computational Perceptual Models

Crowdsourcing enables large-scale perceptual user studies. By using this approach, researchers have gathered large amounts of perceptual data and trained computational perceptual models for quantifying aesthetic assessment [O'Donovan et al. 2011; Secord et al. 2011; Zhu et al. 2014], semantics [Chaudhuri et al. 2013; O'Donovan et al. 2014; Serrano et al. 2016; Streuber et al. 2016; Yumer et al. 2015], and distance metrics [Garces et al. 2014; Liu et al. 2015; Lun et al. 2015; O'Donovan et al. 2014]. (Among them, the method of Zhu *et al.* [2014] may be the most related to ours in that theirs *actively* samples for efficient learning.) Basically, they investigated how to solve *regression* problems for each specific domain. To the best of our knowledge, none of the above studies tries to directly solve *optimization* problems; our Bayesian optimization framework enables active sampling of a design space for efficiently and directly finding an optimum of an unknown perceptual function. Also, we aim at developing an on-demand general solver rather than pre-trained specific models.

3 FUNDAMENTALS OF BAYESIAN OPTIMIZATION

This section briefly introduces the standard Bayesian optimization techniques, upon which our framework is built. We also provide a supplemental document for describing more detailed equations and our implementation. Readers can find more comprehensive introductions in [Brochu et al. 2010b; Shahriari et al. 2016].

Goal. Suppose that $\mathcal{A}$ is a d-dimensional bounded space, $f : \mathcal{A} \to \mathbb{R}$ is an unknown black-box function, and we want to find its maximum:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{A}} f(\mathbf{x}). \tag{1}$$

Suppose as well that the function value $f(\mathbf{x})$ can be computed for an arbitrary point $\mathbf{x}$, but $f(\cdot)$ is an *expensive-to-evaluate* function, *i.e.,* it entails a significant computational cost to evaluate the function value. Thus, while there are many optimization algorithms that can be used here, we are especially interested in making the number of necessary function evaluations as small as possible.

Strategy. Suppose that we currently have a set of t observations:

$$\mathcal{D}_t = \{(\mathbf{x}_i, f_i)\}_{i=1}^{t}, \tag{2}$$

where $f_i = f(\mathbf{x}_i)$. Intuitively, for each iteration of Bayesian optimization, the next evaluation point $\mathbf{x}_{t+1}$ is determined such that

[1] https://www.mturk.com/
[2] https://www.crowdflower.com/

it is "the one most worth observing" based on the previous data $\mathcal{D}_t$. Suppose that $a : \mathcal{A} \to \mathbb{R}$ is a function that quantifies the "worthiness" of the next sampling candidate. We call this function an *acquisition function*. For each iteration, the system maximizes the acquisition function to determine the most effective next sampling point:

$$\mathbf{x}_{t+1} = \underset{\mathbf{x} \in \mathcal{A}}{\arg\max}\ a(\mathbf{x}; \mathcal{D}_t). \tag{3}$$

Acquisition function. We want to choose the next sampling point so that it is likely to have a larger value (because we want to find the maximum) and at the same time its evaluation is more informative (*e.g.,* visiting points that are very close to already visited points is less useful). The *expected improvement* (EI) criterion is often used to ensure such properties [Brochu et al. 2010a, 2007; Snoek et al. 2012]. Let f^+ be the maximum value among the observed data $\mathcal{D}$. The acquisition function based on EI is defined as

$$a^{\mathrm{EI}}(\mathbf{x}; \mathcal{D}) = \mathbb{E}[\max\{f(\mathbf{x}) - f^+, 0\}], \tag{4}$$

where $\mathbb{E}[X]$ means the expectation value of X, and here $f(\cdot)$ is considered as a probabilistic variable that depends on the data $\mathcal{D}$.

Gaussian process prior. It is possible to calculate the expectation in Equation 4 in closed form by assuming a *Gaussian process* (GP) prior on $f(\cdot)$. Under this assumption, any finite set of function value observations follows a multivariate Gaussian distribution. Let $\boldsymbol{\theta}$ be the hyperparameters of this multivariate Gaussian distribution. The GP prior enables an unobserved function value $f(\mathbf{x}_*)$ at an arbitrary point $\mathbf{x}_*$ to be predicted on the basis of the data $\mathcal{D}$ and the hyperparameters $\boldsymbol{\theta}$. It can be analytically proven that the unobserved function value follows a simple Gaussian distribution:

$$f(\mathbf{x}_*) \sim \mathcal{N}(\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)), \tag{5}$$

where $\mu(\cdot)$ is a predicted mean function and $\sigma^2(\cdot)$ is a predicted variance function, both of which can be expressed in closed form.

Hyperparameters. To calculate the predictive distribution of the function value (*i.e.,* $\mu(\cdot)$ and $\sigma(\cdot)$), and consequently the acquisition function $a^{\mathrm{EI}}(\cdot)$, we need to determine the model hyperparameters $\boldsymbol{\theta}$. For this, we use maximum *a posteriori* (MAP) estimation. Given observed data $\mathcal{D}$, the model hyperparameters are determined by maximizing the posteriori distribution of $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{\mathrm{MAP}} = \underset{\boldsymbol{\theta}}{\arg\max}\ p(\boldsymbol{\theta} \mid \mathcal{D}). \tag{6}$$

Example optimization sequence. Figure 3 shows an example sequence of Bayesian optimization with a one-dimensional test function. See the supplementary document for more examples. Note that we do not intend that $\mu(\cdot)$ converges to $f(\cdot)$, since this is not a regression but an optimization; instead, $\mathbf{x}^+$ is expected to converge to the unknown maximum.

4 BAYESIAN OPTIMIZATION BASED ON LINE SEARCH ORACLE

We consider a parameter tweaking process to be a mathematical optimization problem wherein the perceptual (*e.g.,* aesthetic) preference is the objective function to be maximized. Given n parameters to tweak, we solve

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{X}}{\arg\max}\ g(\mathbf{x}), \tag{7}$$

where $g : \mathcal{X} \to \mathbb{R}$ is the *goodness function*, $\mathcal{X} = [0, 1]^n$ is the target *design space*, and $\mathbf{x}^*$ is the optimal parameter set that maximizes the aesthetic preference of the target visual design. As $g(\cdot)$ is an unknown black-box function existing in the human brain, we need to take a human-in-the-loop approach.

The standard Bayesian optimization in the previous section requires that function values can be observed for any argument. In other words, it is based on a *function-value* oracle. However, in our problem setting, it is not realistic to use a function-value oracle for the perceptual goodness function $g(\cdot)$. For example, suppose that you are asked to provide a real-valued goodness score for a certain visual design; this task would be rather difficult without knowing all possible design alternatives. For this reason, Brochu *et al.* [2007] extended Bayesian optimization so that it could use a *function-value-comparison* oracle instead of a function-value oracle; their form of optimization iteratively queries a (human) processor about which design is better in pairwise comparison of two designs.

In this section, we describe a novel extension of Bayesian optimization based on a *line search* oracle instead of function-value or function-value-comparison oracles. The line search oracle is provided by a *single-slider manipulation* query; human processors are asked to adjust a single slider for exploring the design alternatives mapped to the slider and to return the slider value that provides the best design configuration. Mathematically speaking, given a one-dimensional subspace of the entire search space, this oracle provides a solution of a maximization problem within this subspace.

4.1 Slider Space

We let human processors adjust a slider, *i.e.,* find a maximum in a one-dimensional continuous space. We call this space the *slider space*. Technically, this space is not necessarily linear with respect to the target design space $\mathcal{X}$ (*i.e.,* forming a straight line segment in $\mathcal{X}$); however, in this study, we will consider only the case of straight line for simplicity and for the sake of not confusing the human processors.

At the beginning of the optimization process, the algorithm does not have any data about the target design space $\mathcal{X}$ or the goodness function $g(\cdot)$. Thus, for the initial slider space, we simply choose two random points in $\mathcal{X}$ and connect them by a line segment.

For each iteration, we want to arrange the next slider space so that it is as "meaningful" as possible for finding $\mathbf{x}^*$. We propose to construct the slider space $\mathcal{S}$ such that one end is at the *current-best* position $\mathbf{x}^+$ and the other one is at the *best-expected-improving* position $\mathbf{x}^{\mathrm{EI}}$. Suppose that we have observed t responses so far, and we are going to query the next oracle. The slider space for the next iteration, *i.e.,* $\mathcal{S}_{t+1}$, is constructed by connecting

$$\mathbf{x}_t^+ = \underset{\mathbf{x} \in \{\mathbf{x}_i\}_{i=1}^{N_t}}{\arg\max}\ \mu_t(\mathbf{x}), \tag{8}$$

$$\mathbf{x}_t^{\mathrm{EI}} = \underset{\mathbf{x} \in \mathcal{X}}{\arg\max}\ a_t^{\mathrm{EI}}(\mathbf{x}), \tag{9}$$

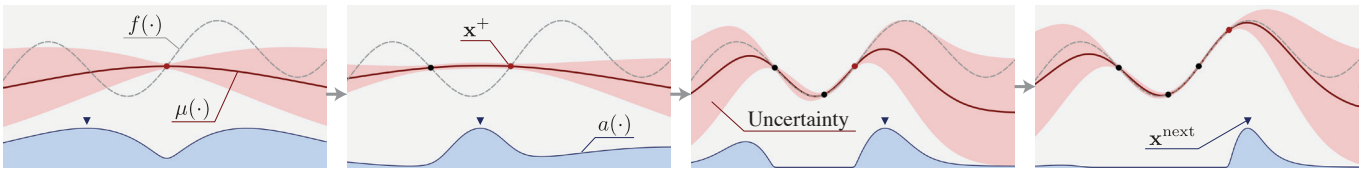

Fig. 3. **An example sequence of Bayesian optimization, applied to a one-dimensional test function.** The iteration proceeds from left to right. The gray dotted line indicates the unknown black-box function $f(\cdot)$, the red line indicates the predicted mean function $\mu(\cdot)$, the blue line indicates the acquisition function $a(\cdot)$, the pink region indicates the uncertainty, *i.e.,* 95% confidence interval, and the dots indicate the observed data (the red one is the maximum at each moment). Note that $a(\cdot)$ is scaled for visualization purpose.

where $\{\mathbf{x}_i\}_{i=1}^{N_t}$ is the set of observed data points, and $\mu_t(\cdot)$ and $a_t^{\text{EI}}(\cdot)$ are the predicted mean function and the acquisition function calculated from the current data. The calculation of $\mu(\cdot)$ and $a^{\text{EI}}(\cdot)$ is less trivial than in the case of standard Bayesian optimization; we will detail it in the following subsections.

Optionally, we can enlarge the line segment with a fixed scale, *e.g.,* 1.25. This is for avoiding cognitive biases; human processors might feel uncomfortable choosing the ends of the slider. Another reason is that the neighborhoods of $\mathbf{x}^+$ and $\mathbf{x}^{\text{EI}}$ are each likely to be good and worth exploring. Moreover, to avoid meaningless slider tweaking, we ensure that the length is not less than 0.25.

4.2 Likelihood of Single-Slider Manipulation Responses

Bradley-Terry-Luce model. In crowdsourced perceptual user studies, pairwise comparison tasks are frequently used. To model pairwise comparison responses from a probabilistic viewpoint, many recent studies (*e.g.,* [O'Donovan et al. 2014; Zhu et al. 2014]) have used the Bradley-Terry (BT) model [Bradley and Terry 1952]. For handling cases in which more than two options are involved, the Bradley-Terry-Luce (BTL) model, which is an extension of the BT model, can be used (see [Tsukida and Gupta 2011]). Suppose that there are m design options corresponding to parameter sets $\mathcal{P} = \{\mathbf{x}_i\}_{i=1}^m$, and the design corresponding to $\mathbf{x}_j$ is chosen out of the m options. We describe this situation as

$$\mathbf{x}_j \succ \mathcal{P} \setminus \{\mathbf{x}_j\}. \tag{10}$$

The BTL model describes the likelihood of this situation as

$$p(\mathbf{x}_j \succ \mathcal{P} \setminus \{\mathbf{x}_j\} \mid \{g_i\}_{i=1}^m) = \frac{\exp(g_j/s)}{\sum_{i=1}^m \exp(g_i/s)}, \tag{11}$$

where g_i denotes the goodness value on $\mathbf{x}_i$, and s is a scaling factor that affects the likelihood; when s is smaller, the likelihood is more sensitive to the goodness function values, and when s is larger, the likelihood becomes closer to $1/m$ and less sensitive to the goodness function values. At present, we consider a fixed value of $s = 0.01$ (but later we will explain how we set this value adaptively in the crowdsourcing setting).

Modeling slider responses. To model the likelihoods of single-slider manipulation responses, we propose to use the BTL model as follows. Let $\mathbf{x}^{\text{chosen}}$ be the parameter set that a human processor chooses from the slider space $\mathcal{S}$ constructed from $\mathbf{x}^+$ and $\mathbf{x}^{\text{EI}}$. Also let $\mathcal{S}'$ be a discretized form of $\mathcal{S}$ consisting of a finite number of points including $\mathbf{x}^{\text{chosen}}$. We describe this situation as

$$\mathbf{x}^{\text{chosen}} \succ \mathcal{S}' \setminus \{\mathbf{x}^{\text{chosen}}\}, \tag{12}$$

and then apply the BTL model by considering that $\mathbf{x}^{\text{chosen}}$ is chosen out of the finite number of options. In our current implementation, we define $\mathcal{S}' = \{\mathbf{x}^{\text{chosen}}, \mathbf{x}^+, \mathbf{x}^{\text{EI}}\}$. We tested several definitions of $\mathcal{S}'$ that included more sampling points, but we did not observe any significant improvement in the optimization behavior; thus, we chose the minimal representation.

4.3 Data Representation

Suppose that we have observed t single-slider manipulation responses so far. We represent this data as

$$\mathcal{D}_t = \{\mathbf{x}_i^{\text{chosen}} \succ \{\mathbf{x}_{i-1}^+, \mathbf{x}_{i-1}^{\text{EI}}\}\}_{i=1}^t, \tag{13}$$

where each $\mathbf{x}_i^{\text{chosen}}$, $\mathbf{x}_{i-1}^+$, and $\mathbf{x}_{i-1}^{\text{EI}}$ corresponds to a certain element in a set of N_t observed data points $\{\mathbf{x}_i\}_{i=1}^{N_t}$. Note that, when a new single-slider manipulation response is added, we merge the same or very close points so that the set $\{\mathbf{x}_i\}_{i=1}^{N_t}$ does not contain any duplicate points.

4.4 Inference from Single-Slider Manipulation Data

Let g_i be the goodness function value at the data point $\mathbf{x}_i$ (*i.e.,* $g_i = g(\mathbf{x}_i)$) and $\mathbf{g}$ be an N-dimensional vector that concatenates the goodness values at all the observed data points:

$$\mathbf{g} = \begin{bmatrix} g_1 & \cdots & g_N \end{bmatrix}^T. \tag{14}$$

Unlike standard Bayesian optimization, in our case, the function values $\mathbf{g}$ are *latent*; they are not explicitly observed, but rather implicitly inferred from the single-slider manipulation responses $\mathcal{D}$. As the goodness values $\mathbf{g}$ and the model hyperparameters $\boldsymbol{\theta}$ are correlated, we infer $\mathbf{g}$ and $\boldsymbol{\theta}$ *jointly* by using MAP estimation:

$$\begin{aligned}(\mathbf{g}^{\text{MAP}}, \boldsymbol{\theta}^{\text{MAP}}) &= \arg\max_{(\mathbf{g},\boldsymbol{\theta})} p(\mathbf{g}, \boldsymbol{\theta} \mid \mathcal{D}) \\ &= \arg\max_{(\mathbf{g},\boldsymbol{\theta})} p(\mathcal{D} \mid \mathbf{g}, \boldsymbol{\theta}) p(\mathbf{g} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}).\end{aligned} \tag{15}$$

Since $\mathcal{D}$ and $\boldsymbol{\theta}$ are conditionally independent given $\mathbf{g}$, we have

$$p(\mathcal{D} \mid \mathbf{g}, \boldsymbol{\theta}) = p(\mathcal{D} \mid \mathbf{g}). \tag{16}$$

The conditional probability $p(\mathcal{D} \mid \mathbf{g})$ is calculated using the BTL model:

$$p(\mathcal{D} \mid \mathbf{g}) = \prod_i p(\mathbf{x}_i^{\text{chosen}} \succ \{\mathbf{x}_{i-1}^+, \mathbf{x}_{i-1}^{\text{EI}}\} \mid \mathbf{g}). \tag{17}$$

Algorithm 1 Bayesian optimization based on line search oracle.

1: **for** $t = 1, 2, \ldots$ **do**
2: $(\mathbf{g}_t^{\mathrm{MAP}}, \boldsymbol{\theta}_t^{\mathrm{MAP}}) = \texttt{compute\_MAP\_estimate}(\mathcal{D}_t)$
3: $\mathbf{x}_t^{+} = \arg\max_{\mathbf{x} \in \{\mathbf{x}_i\}} \mu_t(\mathbf{x})$
4: $\mathbf{x}_t^{\mathrm{EI}} = \arg\max_{\mathbf{x} \in \mathcal{X}} a_t^{\mathrm{EI}}(\mathbf{x})$
5: $\mathcal{S}_{t+1} = \texttt{construct\_slider\_space}(\mathbf{x}_t^{+}, \mathbf{x}_t^{\mathrm{EI}})$
6: $\mathbf{x}_{t+1}^{\mathrm{chosen}} = \texttt{query\_line\_search}(\mathcal{S}_{t+1})$
7: $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{\mathbf{x}_{t+1}^{\mathrm{chosen}} \succ \{\mathbf{x}_t^{+}, \mathbf{x}_t^{\mathrm{EI}}\}\}$
8: **end for**

The conditional probability $p(\mathbf{g} \mid \boldsymbol{\theta})$ is calculated from the definition of the GP prior:

$$p(\mathbf{g} \mid \boldsymbol{\theta}) = \mathcal{N}(\mathbf{g}; \mathbf{0}, \mathbf{K}), \tag{18}$$

where $\mathbf{K}$ is the covariance matrix of this GP, which depends on $\boldsymbol{\theta}$ (see the supplemental document). For $p(\boldsymbol{\theta})$, we assume a log-normal distribution for each hyperparameter (see the supplemental document). As the derivatives can be analytically derived, this MAP estimation can be efficiently performed by gradient-based optimization algorithms such as L-BFGS [Liu and Nocedal 1989].

Once $\mathbf{g}^{\mathrm{MAP}}$ and $\boldsymbol{\theta}^{\mathrm{MAP}}$ have been obtained, we can compute the predictive distribution of the goodness function values for an arbitrary argument, *i.e.,* $\mu(\cdot)$ and $\sigma(\cdot)$, in the same way as in the previous section. Consequently, we can compute the acquisition function $a^{\mathrm{EI}}(\cdot)$.

4.5 Example Optimization Sequence

Algorithm 1 summarizes the procedure of our Bayesian optimization framework. In line 6, the system queries a human. Figure 4 illustrates an example optimization sequence in which the framework is applied to a two-dimensional test function and the oracles are synthesized by a machine processor. The process begins with a random slider space. After several iterations, it reaches a good solution. See the supplemental document and video for more detailed illustrations. Again, as this is not regression, the predicted mean function $\mu(\cdot)$ does not converge to the goodness function $g(\cdot)$, which is the key that enables it to find maximums efficiently.

5 CROWD-POWERED VISUAL DESIGN OPTIMIZER

We define a crowd-powered visual design optimizer as a system that finds an optimal design which maximizes some perceptual function from a given design space and, to enable this, bases its optimization algorithm upon the use of crowdsourced human computation. In this section, we describe the implementation of our crowd-powered visual design optimizer based on the framework described in the previous section.

User experience. We consider a scenario in which a user pushes a "Crowdsource" button in design software for running the crowd-powered optimization process, and then he or she obtains results without any further interaction, as shown in Figure 1. For the user, this seems to be a fully automatic process; indeed, he or she does not need to know that many crowd workers are involved in the computation. Currently, the entire computation takes a few hours

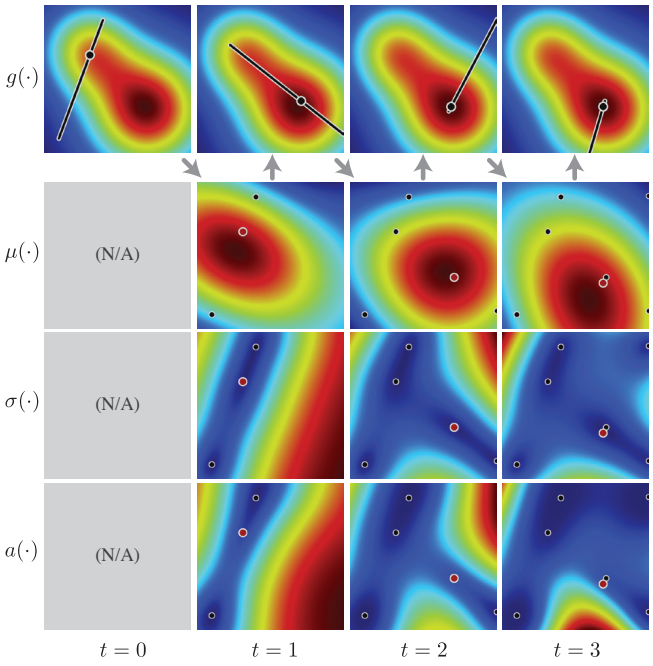

Fig. 4. **An example sequence of the Bayesian optimization based on line search oracle, applied to a two-dimensional test function.** The iteration proceeds from left to right. From top to bottom, each row visualizes the black-box function $g(\cdot)$ along with the next slider space $\mathcal{S}$ and the chosen parameter set $\mathbf{x}^{\mathrm{chosen}}$, the predicted mean function $\mu(\cdot)$, the predicted standard deviation $\sigma(\cdot)$, and the acquisition function $a(\cdot)$, respectively. The red dots denote the best parameter sets $\mathbf{x}^{+}$ among the observed data points at each step.

in our proof-of-concept implementation, and minimization of this latency is out of our scope. Incorporating real-time crowdsourcing techniques [Bernstein et al. 2011] could be used to reduce the latency.

Implementation details. We implemented a microtask platform that crowd workers access through standard web browsers. Instead of generating visual images in real time on web browsers, our platform pre-renders a finite number of images on the server by using uniformly sampled parameter sets along with the slider space. These images are loaded by the crowd workers' web browsers only once when the page is loaded; then the shown image is dynamically updated through slider manipulation in real time. Note that this strategy makes our framework applicable for domains that entail high computational costs for rendering images. To reduce cognitive bias, we set the initial slider tick positions randomly.

Task deployment. We used CrowdFlower as the microtask-based crowdsourcing platform. Other platforms, including Amazon Mechanical Turk, can also be used. We paid 0.05 USD for each task.

Gathering multiple responses. Each crowd worker may respond with some "noise", so averaging the responses from a sufficient number of crowd workers should provide a good approximation of the underlying common preference. To take this into account, we modify the line search query in line 6 in Algorithm 1 as follows. In

each iteration, the system gathers responses from m crowd workers by using the same slider space (*e.g.,* $m = 5$). After gathering the necessary number of responses, the system calculates the median of the provided slider tick positions and uses it for calculating $\mathbf{x}^{\text{chosen}}$. In the actual implementation, we deploy several additional tasks so that there would be no long waits for unrealistically slow workers (*e.g.,* over 30 minutes).

Quality control. Crowd workers might cheat or misunderstand their tasks and thus make poor-quality responses. To detect such low-quality responses, we use a simple quality control approach: we duplicate each task and let each worker do the same task twice, but the slider ends are reversed the second time. If a crowd worker submits contradictory values, *i.e.,* the distance between the slider tick positions is over 25% of the slider length, we consider that he or she is an outlier and ignore the data.

Taking variance in responses into consideration. If there is a perceptually clear maximum in the slider space, crowd workers tend to make similar responses; on the other hand, if there is no clear maximum, crowd workers tend to provide high-variance responses. In the latter case, the data likelihoods should be less influential in the MAP estimation. To make our MAP estimation variance-adaptive, we modify the scale factor s in Equation 11 as $s = a\exp(b\sigma^2)$, where a and b are fixed parameters for controlling the behavior (we used $a = 0.01$ and $b = 50.0$), and σ^2 is the variance of the chosen slider positions, where the width of the slider is taken to be 1.0. When the variance is zero (*i.e.,* all crowd workers provide the same responses), this becomes identical to the unmodified formulation. When the variance is higher, s becomes larger, which means the likelihood calculated by Equation 11 becomes less influential in the MAP estimation. In all the following experiments, we use this modified parameter.

6 EXAMPLE SCENARIOS AND RESULTS

We tested our framework in two typical parameter tweaking scenarios: photo color enhancement and material BRDF design. In both cases, domain-specific approaches are possibly more effective; however, we emphasize that our framework does not rely on any domain knowledge (the application domains are not limited to these two) and thus it can be applied to a wide range of scenarios. In addition, ours can be combined with domain-specific approaches to build more practical specific systems (we leave this for future work).

Costs. All results shown in this section were generated with 15 iterations. For each iteration, our system deployed 7 microtasks, and it proceeded to the next iteration once it had obtained at least 5 responses. We paid 0.05 USD for each microtask execution, so that the total payment to the crowds was 5.25 USD for each result. Typically, we obtained a result in a few hours (*e.g.,* the examples in Figure 5 took about 68 minutes on average).

6.1 Photo Color Enhancement

Photo color enhancement requires adjusting multiple parameters while considering the image contents [Bychkovsky et al. 2011; Koyama et al. 2016]. In this work, we chose the following six parameters for the target design space: brightness, contrast, saturation, and color balance with respect to red, green, and blue, following an existing (publicly available) implementation [Koyama et al. 2016]. Note that our framework can also handle tonal curves by parameterizing them (*e.g.,* [HaCohen et al. 2011]), though we did not use them in this example. In the microtasks, we instructed the crowd workers simply to adjust the slider until the image looked the best.

We compared our optimization with auto-enhancement functions in commercial software packages. Although such enhancement functions heavily utilize domain knowledge, they still may not be sufficiently robust to handle certain classes of photographs (*e.g.,* ones that require semantic interpretation). We compared the results of our enhancement (with 15 iterations) with Adobe Photoshop CC[3] (applying "Auto Tone" and then "Auto Color") and Adobe Photoshop Lightroom CC[4] (setting both "WB" (white balance) and "Tone" to "Auto"). Figure 5 shows the results. To quantify the degree of success of each enhancement is, we conducted a crowdsourced study in which we asked crowd workers to identify which image looks best among the three enhancement results and the original image. For quality control, we duplicated each questionnaire and discarded answers from participants who provided inconsistent answers. The numbers in Figure 5 represent the results. The photos enhanced by our crowd-powered optimization were preferred over the others in these cases. Note that we do not claim that our method is "better" than the other software; instead, these results simply indicate that our method can successfully produce a "people's choice". This represents one of the advantages of our method.

Next, to see the robustness of our framework with respect to varying the initial randomized seeds, we repeated the same optimization procedure three times (Trial A, B, and C). Figure 6 (Top) shows the sequences of enhanced photographs over the iterations. We measured the differences between the trials by using two metrics: a *parameter space metric* and a *perceptual color metric*. The former is based on the l^2-norm in the space $\mathcal{X}$ between the corresponding parameter sets. The latter is based on the perceptual color distance metric called CIEDE2000 [Sharma et al. 2005]; we measured the perceptual distance for each pixel in the enhanced photographs and calculated the mean over all the pixels. Figure 6 (Bottom) shows the results. It shows that the distances become small rapidly in the first 4 or 5 iterations, and they approach similar enhancement even though the initial conditions are quite different.

6.2 Material BRDF Design

Material BRDF design is such a complex and unintuitive task even for experts that special supports are necessary [McAuley et al. 2012; Ngan et al. 2006; Serrano et al. 2016]. Here, we use "Standard Shader" provided in Unity 5[5] as the target design space. This shader provides physically based shading and can be used to express various BRDFs such as plastic, metal, and fabric. In this shader, BRDF is parametrized by albedo lightness, specular lightness, and smoothness. The number of free parameters is three in monotone and seven in full color. When rendering images, we set an HDR skybox and reflective probes to this scene so that the material appearance would be effectively expressed.

[3] http://www.adobe.com/products/photoshop.html
[4] http://www.adobe.com/products/photoshop-lightroom.html
[5] https://unity3d.com/5

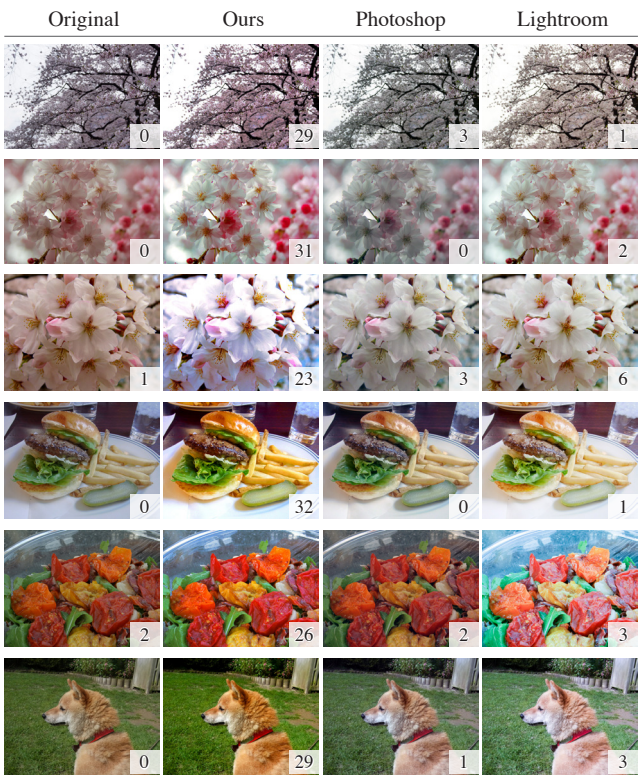

Fig. 5. **Comparison of photo color enhancement between our crowd-powered optimization and auto-enhancement in commercial software packages** (Photoshop and Lightroom). The number on each photograph indicates the number of participants who preferred the photograph to the other three in the study. The second and third photographs are provided by Flickr users: houroumono and Kathleen Conklin.

Our framework enables automatic adjustment of BRDF parameters if a user has a reference photograph; it can be used to minimize the perceptual distance between the appearance in the photograph and the produced appearance by the shader. In the microtasks, we showed both the reference photograph and a rendered image with a slider side by side and asked the crowd workers to adjust the slider until their appearances were as similar as possible. Figure 7 shows the results for both monotone and full color spaces. In a sense, this can be considered to be BRDF acquisition from a casual photograph. This is analogous to the concept of *crowdshaping* [Streuber et al. 2016], while it requires comprehensive perceptual user studies to be done in advance for constructing a model between human shapes and attributes.

Another usage of our framework is that the user can specify textual instructions instead of reference photographs. Figure 8 illustrates the results of this usage, where we instructed crowd workers to adjust the slider so that it looks like "brushed stainless", "dark blue plastic", *etc.* This is not easy when a human-in-the-loop approach is not taken.

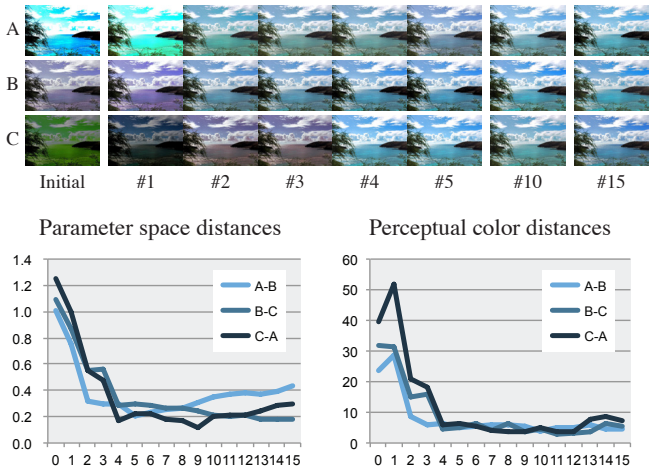

Fig. 6. **Comparison of three optimization trials with different initial conditions in photo color enhancement.** (Top) Transitions of the enhanced images. (Bottom) Transitions of the differences between each trial, measured by the parameter space metric and the perceptual color metric.

7 EVALUATION

In section 6, we showed that our method can produce practical-quality results in two different design domains. The remaining questions to be answered here are: **Q1:** *Does the use of single-slider manipulation (SSM) oracle improve optimization performance?* **Q2:** *How much does the task burden increase as a result of using SSM?* To answer them, we consider the following two baseline conditions. The first is the *2-gallery comparison* (2GC) oracle as used by Brochu *et al.* [2007], where a human processor is asked to choose one option from two given options. The two options are sampled at $\mathbf{x}^{+}$ and $\mathbf{x}^{\mathrm{EI}}$. The second condition is called a *4-gallery comparison* (4GC) oracle, where four (instead of two) options are presented and one option is selected from them. Following [Brochu et al. 2010a], the four options are sampled using Schonlau *et al.*'s method [1998], which is also based on expected improvement and can be seen as a simple extension of 2GC. We modeled the data likelihood in 2GC and 4GC by using the BT and BTL models, respectively. First, we compared our SSM approach with 2GC and 4GC using *synthetic* settings (**Exp1**), where we simulated responses from crowds by using a known test function. Then we compared the three approaches in a *crowdsourcing* setting (**Exp2**). In both **Exp1** and **Exp2**, we evaluated the number of iterations required to get good solutions, to answer **Q1**. In **Exp2**, we also evaluated the microtask burden on the crowds, to answer **Q2**.

7.1 Experiment 1: Synthetic Setting

As a test function to be optimized, we used an n-dimensional function: $g(\mathbf{x}) = \exp\{-(\mathbf{x}-\boldsymbol{\mu})^2/2\sigma^2\}$, where we set $\boldsymbol{\mu} = [\,0.5\ \cdots\ 0.5\,]^T$ and $\sigma = 0.5$. This function has its maximum at $\boldsymbol{\mu}$. We synthesized oracles from this function and tested $n \in \{2, 6, 20\}$. For each iteration, we recorded the residuals: $r = \|\mathbf{x}^{+} - \boldsymbol{\mu}\|$. Figure 9 shows the results. In general, the residuals drop faster at the beginning and converge to smaller values at the end in SSM than in 2GC and 4GC. The

Fig. 7. **Results of the crowdsourced BRDF design with reference photographs.** In each pair, the top image shows the reference photograph and the bottom image shows the resulting BRDF after 15 iterations. The design spaces are 3-dimensional in the left four and 7-dimensional in the right two. Some photographs are provided by Flickr users: Russell Trow, Alexandr Solo, Angie Stalker, Gwen, and lastcun.

Fig. 8. **Results of the crowdsourced BRDF design with textual instructions.** The design spaces are 3-dimensional in the left one and 7-dimensional in the right two.

gallery comparison approaches often provide "flat" graphs where the residual does not decrease for several iterations. The reason for this may be that, especially at the beginning of the iteration, the algorithm is likely to sample the "boundary" of $\mathcal{X}$ because the uncertainty is very large around the boundary. On the other hand, this "boundary-exploration" stage is not a critical problem in the SSM approach, because the slider space lies across the design space even when the one end is on the boundary.

7.2 Experiment 2: Crowdsourcing Setting

To quantify the optimization performance in crowdsourcing settings, we used photo color enhancement with a *reference image*. We manually chose a reference parameter set $\mathbf{x}^{\text{ref}}$ and generated a corresponding image as the ground truth (*i.e.*, a reference image) in advance. In the SSM setting, crowd workers were shown a reference image and an editable image with a slider, and asked to adjust the slider so that the edited image would be as similar to the reference image as possible. In the 2GC and 4GC settings, the crowd workers were shown a reference image and options and asked to find the most similar option. As the quality control in the 2GC and 4GC settings, we duplicated each comparison task while showing the options in opposite order and omitted workers whose responses were contradictory. Figure 10 shows the change in the error as measured by the perceptual color metric over the iterations. We can see that the SSM approach performs better than the 2GC and 4GC approaches do. The trends are mostly consistent with the results of the synthetic settings (Figure 9). Figure 11 visualizes sequences of

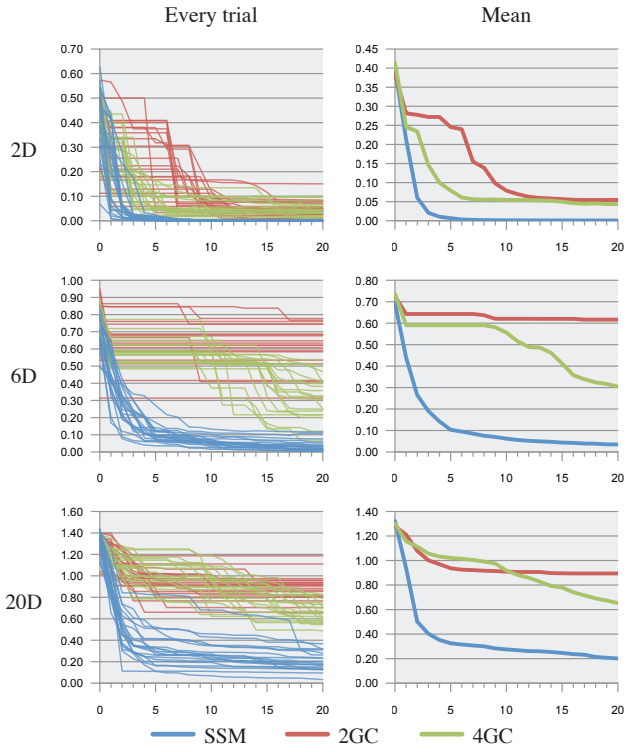

Fig. 9. **Results of the synthetic experiment.** We compare the residuals (vertical axis; lower is better) over iterations (horizontal axis) among the single-slider manipulation (SSM), the 2-gallery comparison (2GC), and the 4-gallery comparison (4GC) settings. We repeated the same procedure 20 times for each condition.

the images enhanced by the predicted best parameter set $\mathbf{x}^{+}$ at each step.

Microtask burden. A possible drawback of the single-slider manipulation task is that it can be more tedious than a comparison task. To respond to this concern, we compared SSM, 2GC, and 4GC in terms of the task-completion time. For each microtask executed by a crowd worker, we measured the elapsed time from the moment that the HTML documents in the task page were loaded to the moment

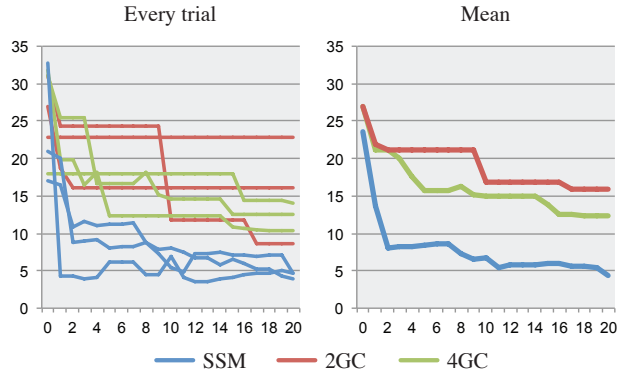

Fig. 10. **Results of the crowdsourcing experiment.** We compare the residuals (vertical axis; lower is better) over iterations (horizontal axis) among the single-slider manipulation (SSM), 2-gallery comparison (2GC), and 4-gallery comparison (4GC) settings. We repeated the same procedure 3 times for each condition.

that the submission button was pushed. The task-completion time included the time for reading the task instructions and the time for conducting duplicate tasks for quality control. Figure 12 shows the results using box plots (the maximum values are not shown for space reasons). It indicates that the SSM microtask requires more time than the other microtasks, but its time is still comparable (less than twice). Considering that the convergence of the SSM approach is much (at least more than two times) faster than those of the others, we argue that our SSM approach is preferable even though the task burden is moderately heavier. Note that this increase of task-completion time does not badly affect the entire latency in practice because other overhead (*e.g.,* between requests of task deployment and findings of the tasks by crowds) is dominant.

8 LIMITATIONS AND FUTURE WORK

Our framework is built upon many assumptions, some of which are difficult to validate quantitatively. For example, we assumed that there exists a common goodness function shared among crowd workers. In Figure 6, we observed that even if the initial parameter sets were different, they eventually converged to similar designs, indicating that this assumption seems valid in this specific situation. However, it is difficult to determine whether this assumption is valid in other situations. For example, there are various color palette styles; some people might prefer a certain style while others might prefer another style. In this case, the assumption of a common goodness function may be invalid.

We have discussed how to reduce the number of queries, but we have not touched on how the time and monetary costs of crowdsourcing can be minimized. To reduce the time cost (*i.e.,* the latency to obtain the optimization results), our framework could incorporate real-time crowdsourcing techniques [Bernstein et al. 2011]. Also, parallelizing (or batching) Bayesian optimization (*e.g.,* [Azimi et al. 2010; Desautels et al. 2014]) may be useful for reducing the entire latency. Reducing the monetary cost may be more challenging. Currently, we always employ a fixed number of crowd workers in each iteration; this number could be adaptively adjusted to each application and each step, but we have not investigated strategies for this. Also, we need to develop a criterion for detecting convergence and thereby stopping the iteration automatically; this will prevent unnecessary tasks from being deployed.

Considering the demographic factors of crowd workers [Reinecke and Gajos 2014] or clustering crowd workers from their responses [Kajino et al. 2013] may enable interesting usage scenarios; these ideas would be worth investigating in the future. For example, the objective of optimization could be customized so that the design is preferred more by a certain group of people.

Our framework does not rely on domain-specific knowledges; this enables it to be used in various design domains. However, to build optimizers for specific design scenarios, it would be more effective to use domain-specific rules or data. Bayesian optimization is capable of taking such domain knowledge into account by incorporating them into the Gaussian process prior (*e.g.,* [Brochu et al. 2010a]).

Investigating the use of the slider-based optimization in a single-user setting also represents a promising area of research. Unlike tweaking raw sliders simultaneously, users do not have to learn and remember the effects of raw parameters; instead, what users have to care about is the best position of the single slider in each step. Considering how user-specific adaptation (*e.g.,* [Koyama et al. 2016]) can be incorporated into our framework is also an interesting topic to explore.

We believe that the single-slider manipulation microtask design could be effective for regression purposes (*e.g.,* [Koyama et al. 2014]) as well. We also expect that our microtask design would be useful for generating a new type of data annotation for machine learning where comparison-based or Likert-scale-based tasks are currently used.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable comments. The photographs provided by Flickr users are licensed under CC BY 2.0 (https://creativecommons.org/licenses/by/2.0/). This work is supported by JSPS KAKENHI under Grant No.: 26240027 and 26-8574. Yuki Koyama is funded by JSPS Research Fellowships for Young Scientists.

REFERENCES

Javad Azimi, Alan Fern, and Xiaoli Z. Fern. 2010. Batch Bayesian Optimization via Simulation Matching. In *Proc. NIPS '10*. 109–117. http://papers.nips.cc/paper/4083-batch-bayesian-optimization-via-simulation-matching.pdf

Michael S. Bernstein, Joel Brandt, Robert C. Miller, and David R. Karger. 2011. Crowds in Two Seconds: Enabling Realtime Crowd-powered Interfaces. In *Proc. UIST '11*. 33–42. DOI: https://doi.org/10.1145/2047196.2047201

Ralph Allan Bradley and Milton E. Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* 39, 3/4 (1952), 324–345. http://www.jstor.org/stable/2334029

Eric Brochu, Tyson Brochu, and Nando de Freitas. 2010a. A Bayesian Interactive Optimization Approach to Procedural Animation Design. In *Proc. SCA '10*. 103–112. DOI: https://doi.org/10.2312/SCA/SCA10/103-112

Eric Brochu, Vlad M. Cora, and Nando de Freitas. 2010b. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. (2010). arXiv:1012.2599.

Eric Brochu, Nando de Freitas, and Abhijeet Ghosh. 2007. Active Preference Learning with Discrete Choice Data. In *Proc. NIPS '07*. 409–416. http://papers.nips.cc/paper/3219-active-preference-learning-with-discrete-choice-data.pdf

Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. 2011. Learning Photographic Global Tonal Adjustment with a Database of Input/Output Image

Fig. 11. **Sequences of the current-best images over iterations in the performance comparison in the crowdsourcing setting.**

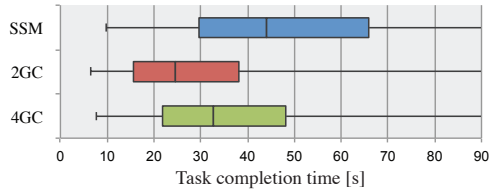

Fig. 12. **Comparison of task-completion times** among single-slider manipulation (SSM), 2-gallery comparison (2GC), and 4-gallery comparison (4GC) microtasks.

Pairs. In *Proc. CVPR '11*. 97–104. DOI: https://doi.org/10.1109/CVPR.2011.5995413

Siddhartha Chaudhuri, Evangelos Kalogerakis, Stephen Giguere, and Thomas Funkhouser. 2013. Attribit: Content Creation with Semantic Attributes. In *Proc. UIST '13*. 193–202. DOI: https://doi.org/10.1145/2501988.2502008

Jeff Clune and Hod Lipson. 2011. Evolving 3D Objects with a Generative Encoding Inspired by Developmental Biology. *SIGEVOlution* 5, 4 (Nov. 2011), 2–12. DOI: https://doi.org/10.1145/2078245.2078246

Thomas Desautels, Andreas Krause, and Joel W. Burdick. 2014. Parallelizing Exploration-Exploitation Tradeoffs in Gaussian Process Bandit Optimization. *Journal of Machine Learning Research* 15 (2014), 4053–4103. http://jmlr.org/papers/v15/desautels14a.html

Elena Garces, Aseem Agarwala, Diego Gutierrez, and Aaron Hertzmann. 2014. A Similarity Measure for Illustration Style. *ACM Trans. Graph.* 33, 4, Article 93 (July 2014), 9 pages. DOI: https://doi.org/10.1145/2601097.2601131

Yotam Gingold, Ariel Shamir, and Daniel Cohen-Or. 2012. Micro Perceptual Human Computation for Visual Tasks. *ACM Trans. Graph.* 31, 5, Article 119 (Sept. 2012), 12 pages. DOI: https://doi.org/10.1145/2231816.2231817

Yoav HaCohen, Eli Shechtman, Dan B. Goldman, and Dani Lischinski. 2011. Non-rigid Dense Correspondence with Applications for Image Enhancement. *ACM Trans. Graph.* 30, 4, Article 70 (July 2011), 10 pages. DOI: https://doi.org/10.1145/2010324.1964965

Hiroshi Kajino, Yuta Tsuboi, and Hisashi Kashima. 2013. Clustering Crowds. In *Proc. AAAI '13*. 1120–1127. http://dl.acm.org/citation.cfm?id=2891460.2891616

Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. 2014. Crowd-powered Parameter Analysis for Visual Design Exploration. In *Proc. UIST '14*. 65–74. DOI: https://doi.org/10.1145/2642918.2647386

Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. 2016. SelPh: Progressive Learning and Support of Manual Photo Color Enhancement. In *Proc. CHI '16*. 2520–2532. DOI: https://doi.org/10.1145/2858036.2858111

Lasse Farnung Laursen, Yuki Koyama, Hsiang-Ting Chen, Elena Garces, Richard Harper Diego Gutierrez, and Takeo Igarashi. 2016. Icon Set Selection via Human Computation. In *Proc. Pacific Graphics 2016 – Short Papers*. 1–6. DOI: https://doi.org/10.2312/pg.20161326

Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller. 2010. TurKit: Human Computation Algorithms on Mechanical Turk. In *Proc. UIST '10*. 57–66. DOI: https://doi.org/10.1145/1866029.1866040

D. C. Liu and J. Nocedal. 1989. On the Limited Memory BFGS Method for Large Scale Optimization. *Math. Program.* 45, 3 (Dec. 1989), 503–528. DOI: https://doi.org/10.1007/BF01589116

Tianqiang Liu, Aaron Hertzmann, Wilmot Li, and Thomas Funkhouser. 2015. Style Compatibility for 3D Furniture Models. *ACM Trans. Graph.* 34, 4, Article 85 (July 2015), 9 pages. DOI: https://doi.org/10.1145/2766898

Zhaoliang Lun, Evangelos Kalogerakis, and Alla Sheffer. 2015. Elements of Style: Learning Perceptual Shape Style Similarity. *ACM Trans. Graph.* 34, 4, Article 84 (July 2015), 14 pages. DOI: https://doi.org/10.1145/2766929

J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. 1997. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. In *Proc. SIGGRAPH '97*. 389–400. DOI: https://doi.org/10.1145/258734.258887

Stephen McAuley, Stephen Hill, Naty Hoffman, Yoshiharu Gotanda, Brian Smits, Brent Burley, and Adam Martinez. 2012. Practical Physically-based Shading in Film and Game Production. In *ACM SIGGRAPH 2012 Courses*. Article 10, 7 pages. DOI: https://doi.org/10.1145/2343483.2343493

Addy Ngan, Frédo Durand, and Wojciech Matusik. 2006. Image-driven Navigation of Analytical BRDF Models. In *Proc. EGSR '06*. 399–407. DOI: https://doi.org/10.2312/EGWR/EGSR06/399-407

Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. 2011. Color Compatibility from Large Datasets. *ACM Trans. Graph.* 30, 4, Article 63 (July 2011), 12 pages. DOI: https://doi.org/10.1145/2010324.1964958

Peter O'Donovan, Jānis Lībeks, Aseem Agarwala, and Aaron Hertzmann. 2014. Exploratory Font Selection Using Crowdsourced Attributes. *ACM Trans. Graph.* 33, 4, Article 92 (July 2014), 9 pages. DOI: https://doi.org/10.1145/2601097.2601110

Alexander J. Quinn and Benjamin B. Bederson. 2011. Human Computation: A Survey and Taxonomy of a Growing Field. In *Proc. CHI '11*. 1403–1412. DOI: https://doi.org/10.1145/1978942.1979148

Katharina Reinecke and Krzysztof Z. Gajos. 2014. Quantifying Visual Preferences Around the World. In *Proc. CHI '14*. 11–20. DOI: https://doi.org/10.1145/2556288.2557052

Matthias Schonlau, William J. Welch, and Donald R. Jones. 1998. *Global versus local search in constrained optimization of computer models*. Lecture Notes–Monograph Series, Vol. 34. Institute of Mathematical Statistics, Hayward, CA, 11–25. DOI: https://doi.org/10.1214/lnms/1215456182

Adrian Secord, Jingwan Lu, Adam Finkelstein, Manish Singh, and Andrew Nealen. 2011. Perceptual Models of Viewpoint Preference. *ACM Trans. Graph.* 30, 5, Article 109 (Oct. 2011), 12 pages. DOI: https://doi.org/10.1145/2019627.2019628

Ana Serrano, Diego Gutierrez, Karol Myszkowski, Hans-Peter Seidel, and Belen Masia. 2016. An Intuitive Control Space for Material Appearance. *ACM Trans. Graph.* 35, 6, Article 186 (Nov. 2016), 12 pages. DOI: https://doi.org/10.1145/2980179.2980242

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* 104, 1 (Jan. 2016), 148–175. DOI: https://doi.org/10.1109/JPROC.2015.2494218

Gaurav Sharma, Wencheng Wu, and Edul N. Dalal. 2005. The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application* 30, 1 (2005), 21–30. DOI: https://doi.org/10.1002/col.20070

Karl Sims. 1991. Artificial Evolution for Computer Graphics. *SIGGRAPH Comput. Graph.* 25, 4 (July 1991), 319–328. DOI: https://doi.org/10.1145/127719.122752

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Proc. NIPS '12*. 2951–2959. http://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf

Stephan Streuber, M. Alejandra Quiros-Ramirez, Matthew Q. Hill, Carina A. Hahn, Silvia Zuffi, Alice O'Toole, and Michael J. Black. 2016. Body Talk: Crowdshaping Realistic 3D Avatars with Words. *ACM Trans. Graph.* 35, 4, Article 54 (July 2016), 14 pages. DOI: https://doi.org/10.1145/2897824.2925981

Kristi Tsukida and Maya R. Gupta. 2011. *How to Analyze Paired Comparison Data*. Technical Report. University of Washington. https://www2.ee.washington.edu/techsite/papers/refer/UWEETR-2011-0004.html

Luis Von Ahn. 2005. *Human Computation*. Ph.D. Dissertation. Carnegie Mellon University, Pittsburgh, PA, USA. Advisor(s) Blum, Manuel. http://reports-archive.adm.cs.cmu.edu/anon/2005/abstracts/05-193.html

Michael J. Wilber, Iljung S. Kwak, and Serge J. Belongie. 2014. Cost-Effective HITs for Relative Similarity Comparisons. In *Proc. HCOMP '14*. http://www.aaai.org/ocs/index.php/HCOMP/HCOMP14/paper/view/8954

Lixiu Yu and Jeffrey V. Nickerson. 2011. Cooks or Cobblers?: Crowd Creativity Through Combination. In *Proc. CHI '11*. 1393–1402. DOI: https://doi.org/10.1145/1978942.1979147

Mehmet Ersin Yumer, Siddhartha Chaudhuri, Jessica K. Hodgins, and Levent Burak Kara. 2015. Semantic Shape Editing Using Deformation Handles. *ACM Trans. Graph.* 34, 4, Article 86 (July 2015), 12 pages. DOI: https://doi.org/10.1145/2766908

Jun-Yan Zhu, Aseem Agarwala, Alexei A. Efros, Eli Shechtman, and Jue Wang. 2014. Mirror Mirror: Crowdsourcing Better Portraits. *ACM Trans. Graph.* 33, 6, Article 234 (Nov. 2014), 12 pages. DOI: https://doi.org/10.1145/2661229.2661287